REMARKS/ARGUMENTS

Preliminary Comments

We thank the Examiner for a telephone interview on May 2, 2007. In that interview, the Examiner confirmed that the rejection of claims 1-6 and 10-15 in paragraph 7 of the Office Action was erroneous, and instead claims 11-14 are rejected under 35 U.S.C § 102. Accordingly, the rejections to claims 11-14 are addressed below.

Amendments to Specification

The title has been amended to address the Examiner's objection to the title.

In paragraph 2 of the detailed action, the Examiner has stated that a new title is required that is clearly indicative of the invention to which the claims are directed and that legal words like "method" and "apparatus" should be removed.

In response, the title has been changed to "TRANSPARENT INTERFACE MIGRATION USING A COMPUTER-READABLE MAPPING BETWEEN A FIRST INTERFACE AND A SECOND INTERFACE TO AUTO-GENERATE AN INTERFACE WRAPPER". It is respectfully submitted that the amended title is representative of the present invention.

35 U.S.C § 102 Claim Rejections

Summary of the Rejections

In paragraph 7 of the Office Action, the Examiner has rejected claims 11-14 under 35 U.S.C. § 102 (b) as being anticipated by the cited Template Software. The Examiner has referred to three Template Software documentation documents in support of the objections of claims 11-14 under 35 U.S.C. § 102 (b), namely: the SNAP communication component (referred to as

COM); the SNAP development environment (referred to as SNAP); and the SNAP permanent storage component (referred to as PERM).

Discussion of Controlling Case Law Decisions

Before setting forth a discussion of the cited art applied in the Office Action, it is useful to set out the relevant test under 35 U.S.C. § 102. "For a prior art reference to anticipate in terms of 35 U.S.C. Section 102 every element of the claimed invention must be identically shown in a single reference." *Diversitech Corp. v. Century Step, Inc.*, 850F. 2d 675, 677, 7 U.S.P.Q. 2d 1315, 1317 (Federal Circuit 1988). "If any claim, element, or step is absent from the reference that is being relied upon, there is no anticipation." *Closter Speedsteel AB v. Crucible, Inc.*, 793F. 2d 1565, 230 U.S.P.Q. 81 (Federal Circuit 1986). The following analysis of the present rejections is respectfully offered with guidance from the foregoing controlling case law decisions.

Summary of Distinguishing Features of the Claimed Invention

The cited art fails to teach or fairly suggest key limitations of the claimed invention. Specifically, the cited art fails to teach or even suggest creating **a computer-readable mapping between a first interface and a second interface**; and **an auto-generator, which uses the computer-readable mapping** to automatically generate an interface wrapper to replace the first interface and allow a software application to use the second interface rather than the first interface.

We will first establish that the Shared Information Based (SIB) connections taught by the Template Software are individually created for each remote process that a local process is intended to communicate with. Accordingly, each SIB connection for each remote process in the Template Software is created without regard to the other remote processes. Hence, a "migration" of the local process from one remote process to a second remote process is done by mapping the local process to each of the remote processes separately, rather than by using a computer-readable mapping between the interface of the first remote process and the interface of the

second remote process to auto-generate an interface wrapper that allows the local process to use the second remote process.

It is respectfully submitted that the above distinction between the Template Software and the claimed invention clearly distinguishes the claimed invention over the teachings of the Template Software.

Analysis of the Cited Art

To begin, as the Examiner points out, the Template Software provides a software development environment in which it is possible to develop applications that include the interaction of two disparate processes. The mechanism that enables two disparate processes to communicate is called a Shared Information Based (SIB) connection. In order for the two processes to communicate they must contain classes that have the same class name, contain the same set of attributes and are defined as shared.

The SIB connection editor can then be used to define which shared classes are being passed over a connection between the two processes. If the shared classes in both of the processes are identical, then the sharing is very straight forward. However, if the shared classes are not identical, i.e. if the shared class in one of the processes contains attributes that are not present in the identically named shared class in the other process, then the attributes that are not common to the two shared classes must be sub-classed in the shared class in the process in which they are present, leaving only the attributes that are common to the two shared classes in the parent class of the sub-class (see COM page 5-5). In the Template Software, the sub-class is referred to as the mapped sub-class and the parent of the mapped sub-class is referred to as the mapped class.

The SIB connection editor can be used to point the attributes of the mapped class to the attributes of the mapped sub-class such that although the sharing of attributes happens between the shared class of one process and the mapped class of the second process, the shared attributes are ultimately stored in the attributes of the mapped sub-class.

SIB connections are defined by two classes, namely the SIB CONNECTION class and the SIB CONNECTION ACCEPTER class. An object of SIB CONNECTION, also called a SIB connection, describes the part of the overall SIB connection that proceeds from the first SNAP process to the second SNAP process. This half of the connection is the half that attempts to establish a connection with the second SNAP process. An object of SIB CONNECTION can only communicate with a single, explicitly defined remote process. If an application needs to communicate with multiple processes, **a separate SIB connection must be defined for each process** (See "The main SIB classes Involved" SNAP page 5-4).

From the foregoing, it is readily apparent that a SIB connection provides a mapping of classes between two processes, i.e. an interface between two processes that allows the sharing of non-identical classes between the two processes. Thus, if a legacy application is currently interacting with a legacy software component/process using a first SIB connection, and the legacy component/process is replaced with a new component/process, a second SIB connection must be defined that includes **a mapping between the legacy application and the new component/process.**

It is important to note that the second SIB connection is based on a mapping between the legacy application and the new component/process. The Template Software **does not teach** that the second SIB connection is based on **a mapping between the interface of the legacy component and the interface of the second process.** Therefore, the Template Software does not disclose a computer-readable mapping from the first interface to the second interface, as recited in the independent claims.

Furthermore, given that the Template Software fails to disclose a computer-readable mapping from the first interface to the second interface, it is clear that the Template Software does not disclose a feature in which an interface wrapper is auto-generated by processing the computer-readable mapping from the first interface to the second interface.

In rejecting claim 11 on page 4 of the Office Action, the Examiner has alleged that PERM 3-28 to 3-29 discloses such a feature. Specifically, the Examiner alleges that "changing the SIB connection and **leaving the mapping in place**-e.g. change from ODBC SIB connection to Oracle" discloses this feature. However, as indicated in the response to the previous Office

Action, it is not clear which type of mapping the Examiner is referring to in this case, as the Examiner has referred to both database mapping and schema mapping in support of the rejection of claim 1. It is assumed that the Examiner is referring to database mapping, as the Examiner's example of changing from an ODBC database to an Oracle database is a change between database connections. However, as indicated in the response to the previous Office Action, a change between database connections would not include the change of a SIB connection, as a database connection rather than a SIB connection is used to interface with databases in the Template Software. Furthermore, it is respectfully submitted that "changing the SIB connection and leaving the mapping in place" clearly does not disclose auto-generating an interface wrapper by processing a computer-readable mapping between a first interface and a second interface. In the example of the database connections, this is clearly illustrated because the first database connection (ODBC database) is set up by mapping the first application to the first database, while the second database connection (Oracle database) is set up by mapping the first application to the second database. There is no teaching of processing a computer-readable mapping between the first database and the second database to generate the second database connection.

In view of the foregoing, it is respectfully submitted that the Template Software fails to teach or fairly suggest the key limitations of independent claims 11 and 14 specified above. Accordingly, it is respectfully submitted that independent claims 11 and 14 distinguish over the teachings of the Template Software, given the rulings in *Closter Speedsteel AB v. Crucible, Inc.* and *Diversitech Corp. v. Century Step, Inc.*

By virtue of their dependency on independent claim 11, it is respectfully submitted that independent claims 12 and 13 distinguish over the teachings of the Template Software for at least the same reasons. It is further submitted that dependent claims 12 and 13, and independent claim 14 recite additional features that distinguish over the Template Software, as described below.

Figures 3A and 3B of the instant application illustrate block diagrams of a legacy application using an interface of a legacy software component, and using an interface wrapper to interact with an interface of a new software component. According to the present invention, the interface wrapper is generated by processing a computer-readable mapping between the interface of the legacy software component (first interface) and the interface of the new software

component (second interface). Thus, during processing, mapping of the classes in the first interface to the classes of the second interface are used to set up the classes of the interface wrapper.

Significantly, it is not the mapping of the classes of the legacy application to the classes of the legacy software component that are used; rather, as stated above, it is the mapping between **the first interface** and **the second interface** that are used to generate the interface wrapper. This is an important distinction. Accordingly, dependent claims 12 and 13, and independent claim 14, recite processing of the classes of the first interface based on the computer-readable mapping between the first interface and the second interface.

In contrast, the Template Software generates a SIB connection by processing the SIB connection set up using the SIB connection editor, which, as described above, is set up by the user to map the classes of a first process to the classes of a second process. If the first process is to communicate with a third process, an additional SIB connection must be defined by the user to map the classes of the first process to the classes of the third process using the SIB connection editor. The additional SIB connection can then be generated using the contents of the SIB editor. However, it is clear that the additional SIB connection **is not auto-generated** by processing a computer-readable mapping between the second process and the third process. Thus the Template Software does not teach or fairly suggest processing of the classes of a first interface using a computer-readable mapping between the first interface and a second interface to generate an interface wrapper for migration of a software application from the first interface to the second interface.

In view of the fact that the Template Software and its associated support documentation fail to teach key limitations of the claims, and also fail to teach every element of the claimed invention, as is required under 35 U.S.C. § 102, given the ruling in *Closter Speedsteel AB v. Crucible, Inc.* and *Diversitech Corp. v. Century Step, Inc.*, the Examiner is respectfully requested to withdraw the 35 U.S.C. § 102(b) rejection of claims 11-14.

35 U.S.C § 103 Claim Rejections

Summary of the Rejections

In paragraph 9 of the Office Action, the Examiner has rejected claims 1-6, 10 and 15 under 35 U.S.C. § 103(a) as being unpatentable over the Template Software in view of "Object-Oriented Information Systems, Planning and Implementation", by David A. Taylor, published April 10, 1992 (Taylor).

In paragraph 11 of the Office Action, the Examiner has rejected claims 7-9 under 35 U.S.C. § 103(a) as being unpatentable over the Template Software in view of Enterprise Application Integration with XML and JAVA, by J.P. Morgenthal *et al.*, 2001, Chapter 5 (XML).

Summary of Distinguishing Features of the Claimed Invention

To begin, Applicant respectfully submits that a first criterion required to establish *prima facie* obviousness has not been satisfied. That is, the combination of the Template Software and Taylor, and the combination of the Template Software and XML do not teach all of the claimed features. In other words, neither Taylor, XML nor the Template Software teaches creating a computer-readable mapping between **a first interface** and **a second interface**; and **an auto-generator, which uses the computer-readable mapping** to automatically generate an interface wrapper to replace the first interface and allow a software application to use the second interface rather than the first interface.

In response to the Examiner's rejections under 35 U.S.C. § 102, we have established above that the Template Software fails to teach or fairly suggest the above feature. We will now establish that the Taylor reference similarly fails to teach or fairly suggest this same feature. This will lead to the conclusion that no combination of the Template Software with either one of, or both of, the Taylor reference and the XML reference would allow one skilled in the art to arrive at the claimed invention.

Analysis of the Cited Art

While it may be true that Taylor explicitly uses the word "wrapper", and teaches the application of the common use of a wrapper in object-oriented software development, Taylor fails to teach or even suggest any implementation aspects of how a "wrapper" is generated. Taylor merely states that a simple strategy employing wrappers involves "placing a wrapper around an entire program. If need be, you can translate messages into commands it understands by feeding it simulated keystrokes, then capture its screen output to formulate a response to the message." (See Taylor, page 297). Taylor also suggests an allegedly more effective (than the prior art) version of the wrapping technique that involves decomposing an existing program into its component functions, and placing an object wrapper around each component function. Taylor states that "[t]hen you can reassemble the components using the message-based interface. The program runs just as before except now it is sending messages back and forth rather than making function calls" (See Taylor, page 297).

From the foregoing, it is clear that Taylor fails to teach or fairly suggest a key limitation of the claimed invention. This key limitation is also not taught or fairly suggested by the Template Software, namely automatically generating an interface wrapper by processing a computer readable mapping between a first interface and a second interface to allow a software application to use the second interface rather than the first interface. Accordingly, it is respectfully submitted that the comments above with regard to the Examiner's rejection of claims 11-14 under 35 U.S.C. § 102 similarly distinguish claims 1-6, 10 and 15 over the cited references.

Given that no combination of the Template Software and Taylor is sufficient to allow one skilled in the art to arrive at the present invention, it is respectfully submitted that the first criteria for a *prima facie* case of obviousness has not been satisfied. Accordingly, it is respectfully submitted that independent claims 1 and 15 distinguish over the cited references and are patentable. It is further submitted that, by virtue of their dependencies on claim 1, dependent claims 2-6 and 10 distinguish over the cited references for at least the same reasons.

The Examiner is respectfully requested to reconsider and withdraw the rejection of claims 1-6, 10 and 15 under 35 U.S.C. § 103(a).

With respect to the XML reference and the rejection of claims 7-9, it is pointed out that Claims 7-9 are dependent on claim 1. As outlined above in response to the other 35 U.S.C. 103(a) rejections, the Template Software and its associated support documentation fails to teach key limitations of claim 1. Specifically, the Template Software fails to teach creating a computer-readable mapping between **a first interface** and **a second interface**; and **an auto-generator, which uses the computer-readable mapping** to automatically generate an interface wrapper to replace the first interface and allow a software application to use the second interface rather than the first interface. As the XML reference also fails to teach these key limitations of claim 1, claim 1 distinguishes over both the Template Software and the XML reference and all claims that depend from claim 1, including claims 7-9, distinguish over the Template Software and the XML reference for at least the same reasons.

Since the Template Software and its associated support documentation and the XML reference fail to teach key limitations of the present invention, the first criteria for the *prima facie* case of obviousness has not been satisfied. Applicant therefore respectfully submits that claims 7-9 are patentable over the Template Software and the XML reference since a case of *prima facie* obviousness cannot be established.

In view of the foregoing, early favorable consideration of this application is earnestly solicited. In the event that the Examiner has concerns regarding the present response, the Examiner is encouraged to contact the undersigned at the telephone number listed below.

Respectfully submitted,

MARTIN SOUKUP

By _____

Philip Lapin
Reg. No. 44,443
Tel.: (613) 232-2486

Date: May 22, 2007

PDL:JFS:sng
Encl.